First Named Inventor: Conor P. Morrison
Application No.: 09/872,257
Filed: June 1, 2001
Customer No.: 22971
Attorney Docket No.: 160421.01
Group Art Unit: 2194
Examiner: Van H. Nguyen
Confirmation Number: 4738
Title: Methods and Systems for Creating and Communicating with Computer Processes

Mail Stop AF
Commissioner for Patents
P.O. Box 1450
Alexandria VA 22313-1450

AMENDMENT

Sir:

In response to the final Office Action mailed 11/02/2005, please amend the above-identified application as follows:

Listing of the Claims begins on page 2 of this amendment.

Remarks begin on page 17 of this amendment.

Listing of the Claims:

1. (Currently Amended) A method for a first process running on a computing device to communicate with a second process, the method comprising:

creating a process table on the computing device, wherein each process in the process table is associated with a process identifier that uniquely identifies the process;

rendering the process table accessible to the first process;

associating a Universally Unique Identifier (UUID) with the second process;

creating an entry for the second process in the process table;

associating the UUID of the second process with the process entry for the second process in the process table;

configuring the second process to respond to a global synchronization event by releasing resources, reporting status, and performing a controlled shutdown;

specifying a communications task to perform; and

using the UUID of the second process to specify that the communications task be performed with respect to the second process.

2. (Original) The method of claim 1 wherein creating a process table comprises creating the process table as shared memory on the computing device.

3. (Original) The method of claim 1 further comprising:

coordinating access to the process table and to the process entry for the second process via software locks.

4. (Original) The method of claim 1 wherein specifying a communications task to perform comprises specifying monitoring a process and wherein the method further comprises:

writing status information about the second process into the process entry for the second process; and

retrieving the status information about the second process by using the UUID of the second process to access the process entry for the second process in the process table.

5. (Original) The method of claim 4 wherein writing status information comprises periodically writing a heartbeat update time and wherein the method further comprises:

comparing the heartbeat update time in the status information to the current time; and

determining if the second process is running based on the comparing of the times.

6. (Original) The method of claim 1 wherein specifying a communications task to perform comprises specifying requesting information from a process and wherein the method further comprises:

specifying a type of information requested; and

returning the information requested to the first process.

7. (Original) The method of claim 6 wherein the type of information requested is selected from the set: log output, console output.

8. (Original) The method of claim 6 further comprising:

specifying a period of time during which to return the information requested; and

wherein returning comprises returning the information requested during the specified period of time.

9. (Original) The method of claim 6 wherein returning comprises returning the information requested until the first process indicates that the information need no longer be returned.

10. (Original) The method of claim 1 wherein specifying a communications task to perform comprises specifying waiting for the second process to achieve a status.

11. (Original) The method of claim 10 wherein the status is in the set: initialized, debug_break, terminated.

12. (Original) The method of claim 1 wherein specifying a communications task to perform comprises specifying sending a signal to the second process.

13. (Original) The method of claim 12 wherein sending a signal indicates that the process should terminate.

14. (Original) The method of claim 1 further comprising:

associating a UUID with a third process;

creating an entry for the third process in the process table;

associating the UUID of the third process with the process entry for the third process in the process table;

associating the UUID of the second process with the process entry for the third process in the process table; and

using the UUID of the second process to specify that the communications task be performed with respect to the third process.

15. (Original) The method of claim 14 wherein the third process is a child of the second process.

16. (Original) The method of claim 15 further comprising using the UUID of the second process to specify that the communications task be performed with respect to all descendents of the second process.

17. (Original) The method of claim 1 wherein the second process runs on a second computing device distinct from the computing device on which the first process runs.

18. (Original) The method of claim 17 further comprising:

associating an identifier of the second computing device with the process entry for the second process in the process table;

creating a second process table on the second computing device;

creating an entry for the second process in the second process table; and

associating the UUID of the second process with the process entry for the second process in the second process table.

19. (Original) The method of claim 18 wherein specifying a communications task to perform comprises specifying monitoring a process and wherein the method further comprises:

writing status information about the second process into the process entry for the second process in the second process table; and

retrieving the status information about the second process by using the UUID of the second process to access the process entry for the second process in the second process table.

20. (Original) A computer-readable medium having instructions for performing the method of claim 1.

21. (Currently Amended) A method for a first process running on a computing device to communicate with a second process and with a third process, the method comprising:

creating a process table on the computing device, wherein each process in the process table is associated with a process identifier that uniquely identifies the process;

rendering the process table accessible to the first process;

creating an entry for the second process in the process table;

creating an entry for the third process in the process table;

associating a group UUID with the process entry for the second process in the process table;

associating the group UUID with the process entry for the third process in the process table;

specifying a communications task to perform; and

using the group UUID to specify that the communications task be performed with respect to the second and third processes.

22. (Original) The method of claim 21 wherein creating a process table comprises creating the process table as shared memory on the computing device.

23. (Original) The method of claim 21 further comprising:

coordinating access to the process table and to the process entries for the second and third processes via software locks.

24. (Original) The method of claim 21 wherein specifying a communications task to perform comprises specifying monitoring a process and wherein the method further comprises:

writing status information about the second process into the process entry for the second process;

writing status information about the third process into the process entry for the third process; and

retrieving the status information about the second and third processes by using the group UUID to access the process entries for the second and third processes in the process table.

25. (Original) The method of claim 24 wherein writing status information comprises periodically writing a heartbeat update time and wherein the method further comprises:

comparing the heartbeat update times in the status information to the current time; and

determining if the second and third processes are running based on the comparing of the times.

26. (Original) The method of claim 21 wherein specifying a communications task to perform comprises specifying requesting information from a process and wherein the method further comprises:

specifying a type of information requested; and

returning the information requested to the first process.

27. (Original) The method of claim 26 wherein the type of information requested is selected from the set: log output, console output.

28. (Original) The method of claim 26 further comprising:

specifying a period of time during which to return the information requested; and

wherein returning comprises returning the information requested during the specified period of time.

29. (Original) The method of claim 26 wherein returning comprises returning the information requested until the first process indicates that the information need no longer be returned.

30. (Original) The method of claim 21 wherein specifying a communications task to perform comprises specifying waiting for the second process to achieve a status and the third process to achieve the status.

31. (Original) The method of claim 30 wherein the status is in the set: initialized, debug_break, terminated.

32. (Original) The method of claim 21 wherein specifying a communications task to perform comprises specifying waiting for the second process or the third process to achieve a status.

33. (Original) The method of claim 32 wherein the status is in the set: initialized, debug_break, terminated.

34. (Original) The method of claim 21 wherein specifying a communications task to perform comprises specifying sending a signal to the second and the third processes.

35. (Original) The method of claim 34 wherein sending a signal indicates that a process should terminate.

36. (Original) The method of claim 21 wherein the second process runs on a second computing device distinct from the computing device on which the first process runs.

37. (Original) The method of claim 36 further comprising:

associating an identifier of the second computing device with the process entry for the second process in the process table;

creating a second process table on the second computing device;

creating an entry for the second process in the second process table; and

associating the group UUID with the process entry for the second process in the second process table.

38. (Original) The method of claim 37 wherein specifying a communications task to perform comprises specifying monitoring a process and wherein the method further comprises:

writing status information about the second process into the process entry for the second process in the second process table; and

retrieving the status information about the second process by using the group UUID to access the process entry for the second process in the second process table.

39. (Original) A computer-readable medium having instructions for performing the method of claim 21.

40-65. (Cancelled)

66. (Currently Amended) A computer-readable medium having instructions for performing steps comprising:

executing a first process in a first computing device;

launching a second process in a second computing device, the second process being invoked by the first process;

identifying the second process in a shared memory included in the first computing device, the second process being identified in the shared memory with an identifier unique to the second process, the unique identifier being independent from the computing device on which the second process is running;

configuring the first process and the second process to communicate based, at least in part, on the information in the shared memory;

associating a global synchronization event with the second process; and

configuring the second process to respond to the global synchronization event by releasing resources, reporting status, and performing a controlled shutdown.

67. (Previously Presented) The computer-readable medium as recited in claim 66, further comprising configuring the second process to periodically log heartbeat entries in the shared memory.

68. (Currently Amended) The computer-readable medium as recited in claim 67, further comprising configuring the first process to access the heartbeat entries logged by the second process in [[the]] a process table.

69. (Currently Amended) The computer-readable medium as recited in claim 66, further comprising:

associating a set of processes with [[an]] a parent identifier that identifies a parent process from which the processes in the set depend; and

in response to a termination of the parent process, canceling the processes identified by the parent identifier.

70. (Currently Amended) A computing device comprising:

means for executing local processes on the computing device;

means for launching remote processes invoked by the local processes on other remote computing devices;

means for maintaining a process table on the computing device that includes information about the local processes and the remote processes, wherein each process in the process table is associated with a process identifier that uniquely identifies the process;

means for identifying the local processes and the remote processes with identifiers that do not distinguish the remote processes from the local processes;

means for enabling the remote processes to update the information in the process table; and

means for the local processes to access the updated information about the remote processes.

71. (Currently Amended) The computing device as recited in claim 70, further comprising means for associating a global synchronization event with each remote process.

72. (Currently Amended) The computing device as recited in claim 71, wherein the global synchronization event includes at least one of means for releasing resources, means for reporting status, and means for performing a controlled shutdown.

73. (Currently Amended) The computing device as recited in claim 70, further comprising:

means for defining global synchronization events associated with all of the remote processes identified in the process table; and

means for assigning meanings to the defined global synchronization events.

74. (Previously Presented) The computing device as recited in claim 70, wherein the information in the process table associated with each remote process includes at least one of a parent identifier, a group identifier, a creation time, a heartbeat time, a source device, a target device, or a process type.

REMARKS

Claims 1-39 and 66-74 were previously pending in this application. In this response, claims 1, 21, 66, and 68-73 are amended. No claims are canceled. Claims 1-39 and 66—74 remain pending.

INTERVIEW SUMMARY

The Applicant thanks the Examiner for the in-person interview conducted November 16, 2005, and attended by the Examiner and Applicant's representatives Andrew D. Enfield and James R. Banowsky. During the interview the Examiner suggested that claim 1 would be allowable if it incorporated all of the limitations of dependent claim 19. Also during the interview, differences between the claims and the teachings of U.S. Patent No. 5,748,489 to Beatty et al. were discussed, including the terms "process table" and "global event".

CLAIM OBJECTIONS

The Office Action objected to claim 69 because it referred to "an parent identifier" instead of "a parent identifier". Applicant has amended claim 69 to correct this typographical error without narrowing the scope of the claim. Accordingly, the objection thereof should be withdrawn.

35 U.S.C. § 103 REJECTIONS

Claims 1-20 and 66-74

Claims 1-20 and 66-74 stand rejected under 35 U.S.C. 103(a) as being unpatentable over U.S. Patent No. 5,748,489 to Beatty et al. (hereinafter "Beatty") in view of U.S. Patent No. 5,706,516 to Chang et al. (hereinafter "Chang"). The applicant respectfully traverses these rejections.

As stated in the specification, the presently claimed subject matter is generally concerned with providing

> mechanisms for creating and communicating with computer processes. An application programming interface (API) presents the services of the invention to applications. The API is usable with all processes, local and remote, and is transparent with respect to the location of processes. The invention also works with processes that do not use the API, although some enhanced services are available only to processes using the API.
>
> A process table stores information about processes created using the invention. The process table is accessible by all processes, local and remote, and supports centralized process control and peer-to-peer process communication and synchronization. Locks are used to synchronize access to the process table.
>
> Each process is assigned a Universally Unique Identifier (UUID) that uniquely identifies the process no matter the computing device on which it runs. A parent UUID and a group UUID may be attached to the process and used for enforcing dependencies (e.g., for waiting for or halting the process and all of its child processes) and for managing arbitrary, user-defined groups, respectively.
>
> A global event is associated with each process. When a process receives this event, it performs a controlled shutdown, cleans up, and reports its status. Users define other global events and assign meanings to them. Global events form a generally useful message-passing mechanism.
>
> At frequent intervals, processes and process threads log heartbeat entries in the process table. If a process or thread stops updating

> this field, then other processes can assume that this process or thread broke into the debugger. A process may log other information such as the number of its threads and the current status of the threads.
>
> Specification, page 2, line 8 to line 30

Beatty describes techniques for partitioning complex circuit analysis tasks into subtasks associated with scheduling (master) and execution (slave) "processes." Chang describes techniques for executing requests on multiple computing nodes via a "fast communication manager."

Claim 1

Claim 1, as amended, recites "a method for a first process running on a computing device to communicate with a second process, the method comprising: creating a process table on the computing device, wherein each process in the process table is associated with a process identifier that uniquely identifies the process; rendering the process table accessible to the first process; associating a Universally Unique Identifier (UUID) with the second process; creating an entry for the second process in the process table; associating the UUID of the second process with the process entry for the second process in the process table; configuring the second process to respond to a global synchronization event by releasing resources, reporting status, and performing a controlled shutdown; specifying a communications task to perform; and using the UUID of the second process to specify that the communications task be performed with respect to the second process."

Claim 1 contains multiple limitations, including the creation of a process table and the particular use of a global synchronization event, that distinguish it over the references of record.

The "process table" described by claim 1 is not the same as the "routing table" of Beatty. The routing table of Beatty describes "where to route information destined for the leaves (i.e., slave processes) in the subtree routed at the master process" (Beatty, column 8, lines 19-22). Based on the descriptions of the routing table contained, for example, in columns 7 and 8 of Beatty, such a routing table appears to be used to track connections between entities. For example, "... when master process 4 receives a response from all its children (e.g., slave process 4), it uses the replacement relation M4<-S4 to generate the connectivity relation S4.a.P4==S4.P3. Additionally, master process 4 builds a routing table: S4 via S4, which in this example is degenerate. Although in this example the routing table is degenerate, it is convenient to have entries in the routing table, even degenerate entries. Thus, if a search fails to find an entry in the routing table, then it knows that an error has occurred" (Beatty, column 7, lines 29-38).

In contrast, the process table of claim 1 "stores information about processes created using the invention" (Specification, page 2, line 14). These processes and the stored information are not the same as the routing and connection information stored in the routing table of Beatty. An exemplary process table is described in the specification in Figure 4. The data stored in such an exemplary process table is not contained in the routing table of Beatty.

As a further example of how the process table of claim 1 is not the same as the routing table of Beatty, consider the differences in the definition of "process" between Beatty and the claimed invention. The "process" of Beatty describes a logical entity for

dividing tasks, rather than a physical process running on a computing device that is associated with metadata like a process identifier. The specification states that "each computing device contains a process table that has an entry for each process running on, or invoked by a process running on, the computing device" (Specification, page 5, lines 7-8). The routing table of Beatty does not contain "an entry for each process running on, or invoked by a process running on, the computing device." Instead, the routing table of Beatty describes connection information, as discussed above. For example, consider Figure 7 of Beatty, which shows 10 master and slave "processes." As described by Beatty, the routing table corresponding to these master and slave processes contains fewer than five items (the creation of the routing table is described in columns 6-8, especially from line 25 of column 7 to line 3 of column 8). If the "processes" of Beatty were the same as the "processes" of claim 1, and the routing table of Beatty was the same as the process table of claim 1, the routing table in Beatty would need to contain an item for every process - that is, it would need to contain 10 items, because Figure 7 of Beatty shows 10 "processes." However, the routing table does not contain 10 items - it instead contains five (or fewer) items - because the items in the routing table of Beatty are not the same as the items in the process table of claim 1, and the routing table of Beatty is not the same as the process table of claim 1.

While Applicant believes it is clear that the process table of claim 1 is different in multiple ways from the routing table of Beatty, claim 1 has been amended to clarify the meaning of "process table" by reciting that "each process in the process table is associated with a process identifier that uniquely identifies the process," as is described in the specification.

As another example of an element distinguishing claim 1 from the references of record, consider the amended limitation of "configuring the second process to respond

to a global synchronization event by releasing resources, reporting status, and performing a controlled shutdown."

The Office Action cites "the events discussion beginning at col. 8, line 32" of Beatty as anticipating the global event of claim 1. Applicant has reviewed this section of Beatty, as well as the other cited references, and can find no discussion of a global synchronization event, let alone an event that is associated with a process "releasing resources, reporting status, and performing a controlled shutdown."

One general dictionary definition of the word "event" is "something that takes place; an occurrence." The events of Beatty described in column 8, starting at line 32, appear to be events in this sense – they are occurrences described in the reference. In contrast, as will be appreciated by those skilled in the art, the global event of claim 1 refers to a computer-oriented global synchronization event used to synchronize multiple threads or processes. For example, in an exemplary implementation, one thread in a process may "wait" on a global event. When that global event is "signaled," possibly by another thread, the waiting thread may continue. Further, the global event of claim 1 is associated with a process that, in response to the signaling of the global event, "releas[es] resources, report[s] status, and perform[s] a controlled shutdown."

While the Applicant believes the previous description of a global event provided by claim 1 sufficiently describes this element, Applicant has amended claim 1 to recite "a global synchronization event," clarifying that a global event is a global synchronization event, as described in the specification and recognized by those skilled in the art. Beatty does not disclose a global synchronization event, let alone a global synchronization event that is associated with a process "releasing resources, reporting status, and performing a controlled shutdown."

Accordingly, for at least these reasons, claim 1 is allowable over the cited combination of references and the rejection thereof should be withdrawn.

Claims 2-20

Claims 2-20 depend from claim 1 and are allowable at least by virtue of this dependency. Accordingly, the rejections of these claims should be withdrawn.

Claim 66

Claim 66, as amended, recites "a computer-readable medium having instructions for performing steps comprising: executing a first process in a first computing device; launching a second process in a second computing device, the second process being invoked by the first process; identifying the second process in a shared memory included in the first computing device, the second process being identified in the shared memory with an identifier unique to the second process, the unique identifier being independent from the computing device on which the second process is running; configuring the first process and the second process to communicate based, at least in part, on the information in the shared memory; associating a global synchronization event with the second process; and configuring the second process to respond to the global synchronization event by releasing resources, reporting status, and performing a controlled shutdown."

Claim 66 has been amended to clarify that the global event referred to by the claim is an event used for synchronization of multiple threads or processes, as is

discussed above in the response to the rejection of claim 1. The cited combination of references fails to show a "global synchronization event" as is described by claim 66.

Accordingly, for at least this reason, claim 66 is allowable over the cited combination of references and the rejection thereof should be withdrawn.

Claims 67-69

Claims 67-69 depend from claim 66 and are allowable at least by virtue of this dependency. Accordingly, the rejections of these claims should be withdrawn.

In addition, claim 68 has been amended to correctly specify antecedent basis, without further narrowing the scope of the claim.

Claim 70

Claim 70, as amended, recites "A computing device comprising: means for executing local processes on the computing device; means for launching remote processes invoked by the local processes on other remote computing devices; means for maintaining a process table on the computing device that includes information about the local processes and the remote processes, wherein each process in the process table is associated with a process identifier that uniquely identifies the process; means for identifying the local processes and the remote processes with identifiers that do not distinguish the remote processes from the local processes; means for enabling the remote processes to update the information in the process table; and means for the local processes to access the updated information about the remote processes."

Claim 70 has been amended to clarify the meaning of "process table" by reciting that "each process in the process table is associated with a process identifier that uniquely identifies the process." As is discussed above in the response to the rejection of claim 1, the "process table" of claim 70 is different from the "routing table" of Beatty.

Accordingly, for at least this reason, claim 70 is allowable over the cited combination of references and the rejection thereof should be withdrawn.

Claims 71-74

Claims 71-74 depend from claim 70 and are allowable at least by virtue of this dependency.

In addition, claims 71-74 have been amended to clarify that the global events referred to by the claims are events used for synchronization of multiple threads or processes, as is discussed above in the response to the rejection of claim 1. The cited combination of references fails to show the "global synchronization events" that are described by claims 71-74.

Accordingly, for at least these reasons, claims 71-74 are allowable over the cited combination of references and the rejections thereof should be withdrawn.

Claims 21-39

Claims 21-39 are rejected under 35 U.S.C. 103(a) as being unpatentable over Beatty in view of Chang and further in view of Bala et al. ("Process groups: a mechanism

for the coordination and communication among processes in the Venus collective communication library," 1993) (hereinafter "Bala").

As discussed above, Beatty describes techniques for partitioning complex circuit analysis tasks into subtasks associated with scheduling (master) and execution (slave) "processes," and Chang describes techniques for executing requests on multiple computing nodes via a "fast communication manager." Bala describes a "process group" abstraction to enable dynamic groups of processes to communicate and coordinate work.

Claim 21

Claim 21, as amended, recites "a method for a first process running on a computing device to communicate with a second process and with a third process, the method comprising: creating a process table on the computing device, wherein each process in the process table is associated with a process identifier that uniquely identifies the process; rendering the process table accessible to the first process; creating an entry for the second process in the process table; creating an entry for the third process in the process table; associating a group UUID with the process entry for the second process in the process table; associating the group UUID with the process entry for the third process in the process table; specifying a communications task to perform; and using the group UUID to specify that the communications task be performed with respect to the second and third processes."

Claim 21 has been amended to clarify the meaning of "process table" by reciting that "each process in the process table is associated with a process identifier that uniquely identifies the process." As is discussed above in the response to the rejection

of claim 1, the "process table" of claim 21 is different from the "routing table" of Beatty. Further, the addition of a process group abstraction, as disclosed by Bala, does not overcome the analysis presented in the previous response to the rejection of claim 1 because Bala does not teach or suggest the elements shown to be absent under the previous analysis.

Accordingly, this claim is allowable over the cited combination of references for at least this reason and the rejections of this claim should be withdrawn.

Claims 22-39

Claims 22-39 depend from claim 21 and are allowable at least by virtue of this dependency. Accordingly, the rejections of these claims should be withdrawn.

Response to Arguments

The "Response to Arguments" section of the Office Action states in paragraph 43 that "Applicant is arguing against the references individually" and that "one cannot show nonobviousness by attacking references individually where the rejections are based on combinations of references." Applicant asserts that the responses discuss why the combination of references fail to show one or more elements of the claims, by showing that certain required elements are lacking from the individual references. When all cited references fail to show one or more elements of the claims, it is appropriate to discuss where in the particular cited reference used by the Office Action the claim elements are not shown.

CONCLUSION

Accordingly, in view of the above amendment and remarks it is submitted that the claims are patentably distinct over the prior art and that all the rejections to the claims have been overcome. Reconsideration and reexamination of the above Application is requested. Based on the foregoing, Applicants respectfully requests that the pending claims be allowed, and that a timely Notice of Allowance be issued in this case. If the Examiner believes, after this amendment, that the application is not in condition for allowance, the Examiner is requested to call the Applicant's agent at the telephone number listed below.

If this response is not considered timely filed and if a request for an extension of time is otherwise absent, Applicants hereby request any necessary extension of time. If there is a fee occasioned by this response, including an extension fee that is not covered by an enclosed check please charge any deficiency to Deposit Account No. 50-0463.

Respectfully submitted,

Microsoft Corporation

Date: January 3, 2006

By: 

Microsoft Corporation
One Microsoft Way
Redmond, WA 98052-6399

Andrew D. Enfield, Reg. No.: 57,651
Agent for Applicants
Direct telephone (425) 703-8227

James R. Banowsky, Reg. No.: 37,773

CERTIFICATE OF MAILING OR TRANSMISSION
(Under 37 CFR § 1.8(a))

I hereby certify that this correspondence is being electronically deposited with the USPTO via EFS-Web on the date shown below:

January 3, 2006
Date


Signature

Rimma N. Oks
Printed Name

Type of Response: Final Response
Application Number: 09/872,257
Attorney Docket Number: 160421.01
Filing Date: 06/01/2001